Verifying Infinite-State Systems with Automata

Axel Legay

Université de Liège

Introduction

Goal

- Verifying Reachability Properties (mutual exclusion, ...) of Infinite-State systems.

Solution

1. Encode the reachability property as a set of states.

2. Compute the set of reachable states ϕ.

3. Use set-based operations (union, intersection,...) to test whether ϕ satisfies the property.

Computing the Reachable Set of States

Problems:

1. Infinite sets of states must be represented by finite structures.

2. Classical operations like union, intersection, and testing inclusion must be computable in an efficient way.

3. Infinite sets of states must be computable in finite time.

   "How to compute the effect of an unbounded number of applications of the transition relation?"

Solution: The Regular Model Checking approach.

The Regular Model Checking Framework

[Pnueli et al. 97], [Boigelot, Wolper, 98], [Bouajjani, Jonsson, Nilsson, Touili, 00]

A Solution to Problems 1 and 2.

The system is given by a triple $M = (\Sigma, I, R)$, where

- Σ is a finite alphabet, over which states are encoded as finite-words;
- I is a set of initial states encoded by a finite-word automaton A_I;
- R is the transition relation represented either by a finite-word transducer T (automaton over $\Sigma \times \Sigma$), or by transformations on automata.

Don't think of automata as programs, but as a Symbolic representation of sets of states/transitions that has a very simple semantics.

Example 1: Parametric Systems (1)

- Systems composed of an unbounded number of identical finite-state processes. This number is fixed during an execution.
- Each state is encoded by a finite word.
- Each letter of the word encodes the current state of a process.
- Sets of states are represented by finite-word automata.
- Linear and ring topologies can be encoded in this framework.

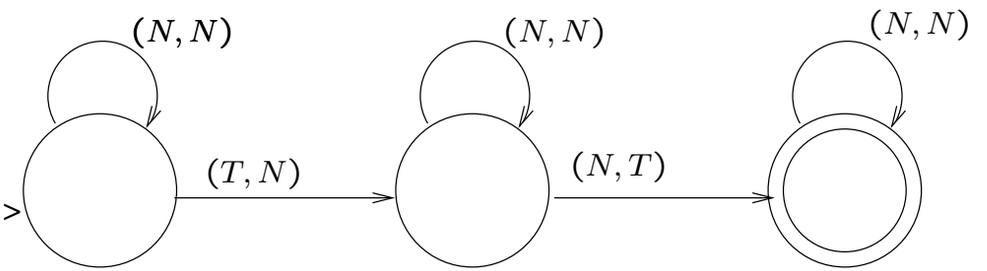
Parametric Systems (2)

Consider a parametric token ring where each process can be in two states: N when it wants the token and T when it has the token,

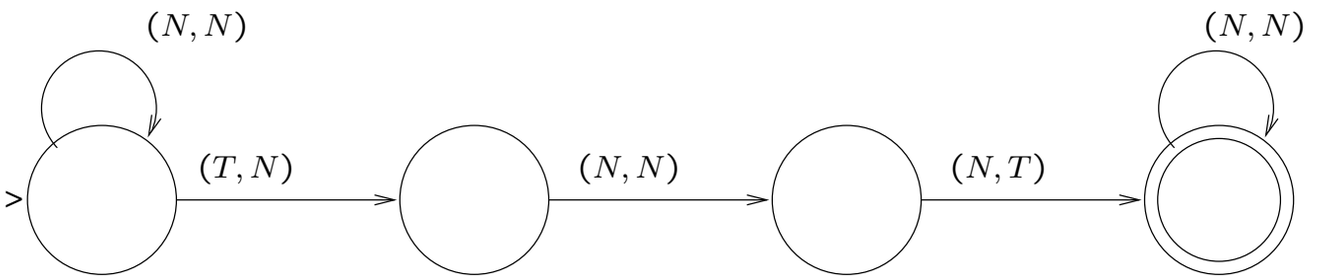
- $TNNNN$ is a state in where the first process has the token.
- $I = TN^*$ models all possible states in where the first process has the token.
- $T = (N, N)^*(T, N)(N, T)(N, N)^*$ is a regular relation that represents the token passing from a process to its right neighbor.

Parametric Systems (3)

Transducer T for $(N,N)^*(T,N)(N,T)(N,N)^*$:

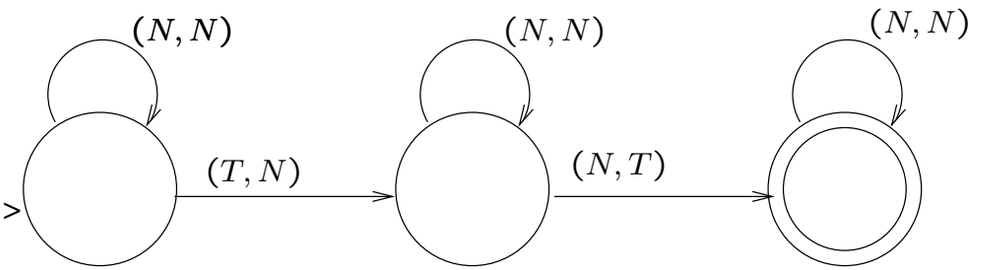


Transducer for $T^2 = T \circ T$:

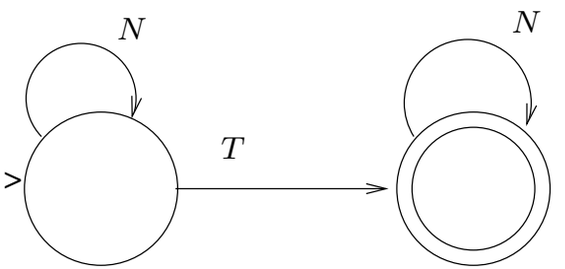


Parametric Systems (3)

Transducer for $T^* = \bigcup_{i \geq 0} T^i$:

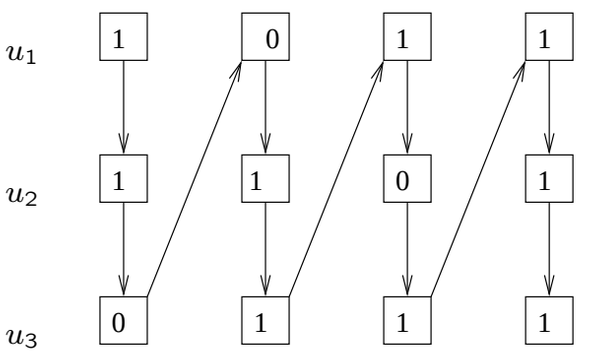


Automaton for $T^*(I)$:



Example 2: Integer systems

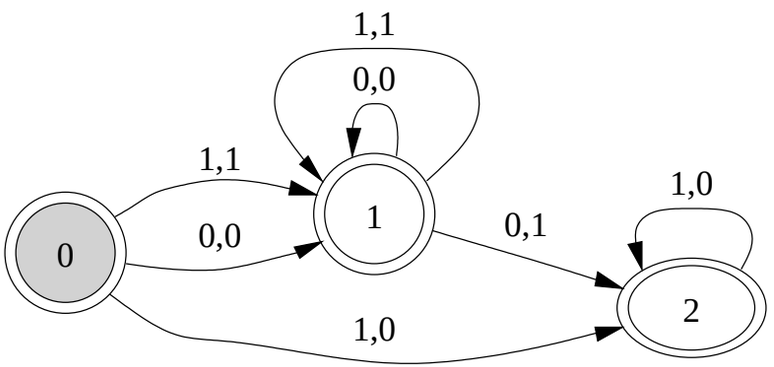
- For arithmetic, integers are encoded in binary, using 2's complement for negative numbers.
- All possible encodings are considered, i.e. 01, 001, ...
- For vectors, same-position bits of the components are read simultaneously.



- With this encoding all Presburger definable relations (and more) are representable [Boigelot, 99].

Arithmetic transducers: Example

Transducer for $(x, x+1) \cup (x, x)$:



Beyond Regular Model Checking

Other applications include

- Communication protocols.
- Recursion.
- Heap Analysis.

Next challenge.

- An important class cannot be encoded: Timed Systems.
- Solution: using infinite words and Büchi automata as a regular representation of sets of states.

Omega-Regular Model checking

- We aim at using Büchi automata as a representation for sets of states.

- The main difficulty is that there are many operations which are not easily applied to the full class of Büchi automata (example: complementation).

- We thus restrict ourself to a convenient sub-class of infinite word languages, those accepted by weak deterministic Büchi automata [Boigelot, Legay, Wolper, 04].

- Weak deterministic Büchi automata behave like finite-word automata.

The Omega Regular Model Checking Framework

A system is a triple $S = (\Sigma, I, R)$ where

- Σ is a finite alphabet, over which the system states are encoded as infinite words;
- I is a set of initial states represented by a weak deterministic automaton A_I;
- R is a transition relation represented by a weak deterministic automaton over $\Sigma \times \Sigma$, i.e. a weak deterministic transducer T.

(ω-)Regular Model Checking Problems

- The goal is to compute $T^* = \bigcup_{i \geq 0} T^i$. This provides a solution to Problem 3.

Two main approaches have been followed.

- Specific techniques that provide a solution for a specific class of systems.

- Generic techniques that operate directly on the automata-based representation without any assumption on the system being considered.

Next slides: The Generic Technique developed at Liège.

The Generic Techniques developed at Liège

Computing $T^* = \bigcup_{i \geq 0} T^i$ is equivalent to compute the limit of

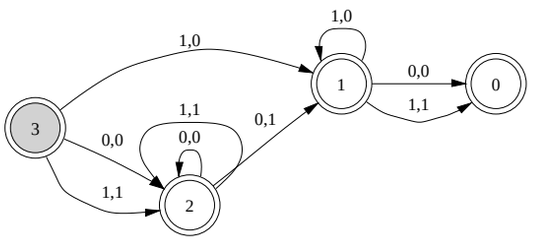
$$T, T^2, T^3, \ldots,$$

Approach:

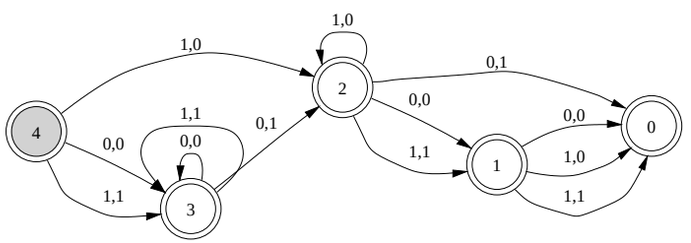
- Sampling: Only consider selected powers of the sequence to be iterated.
- Increment detection and extrapolation: Proceed by comparing successive REDUCED automata in a sequence; identifying the difference between these in the form of an "increment", and extrapolating the repetition of this increment.
- Preciseness check: Check algorithmically whether the extrapolated sequence is "exact".

Increments: Example

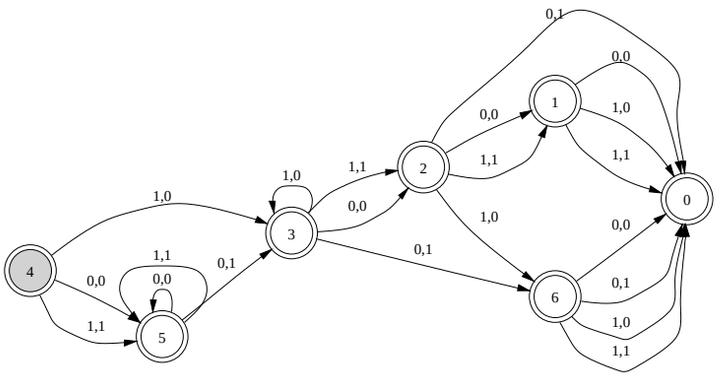
T^2:



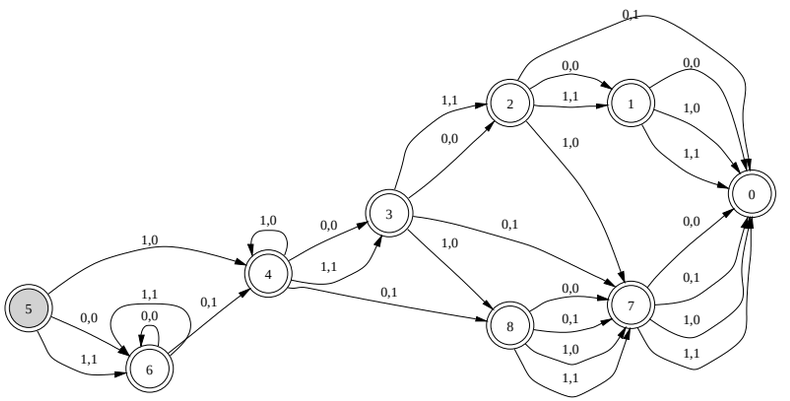
T^4:



T^8:



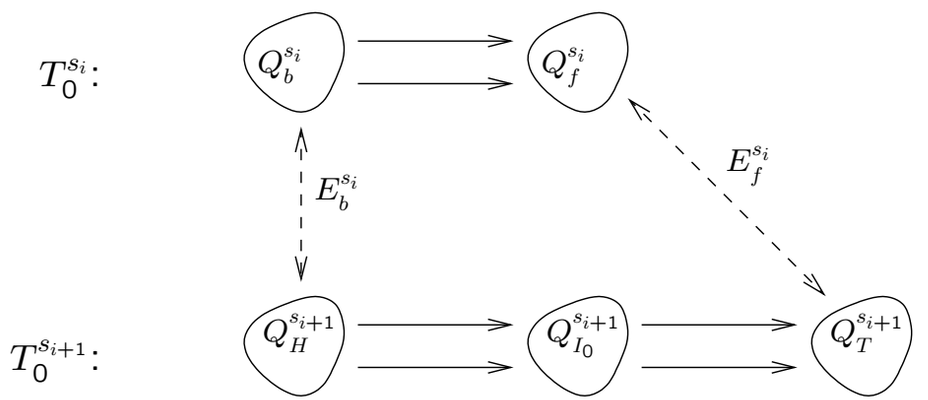
T^{16}:



Detecting increments

Common parts between $T_0^{s_i}$ and $T_0^{s_i+1}$ can be identified by

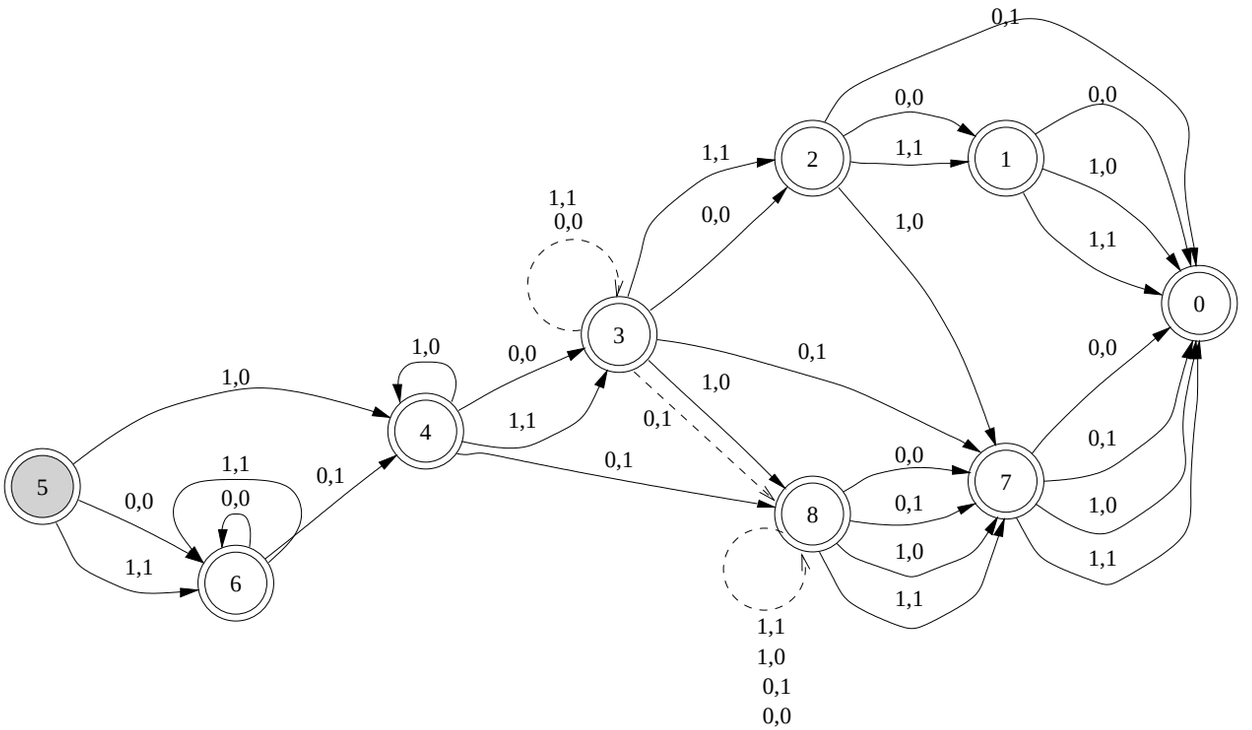
- a *forwards equivalence relation* $E_f^{s_i}$ between states, based on equality of accepted languages, and
- a *backwards equivalence relation* $E_b^{s_i}$, based on isomorphism from the initial state.



This detection step is propagated to several increments.

Extrapolation: Example

Extrapolation = LOOP!



Increment $\{3, 8\}$ is repeated an unbounded number of times.

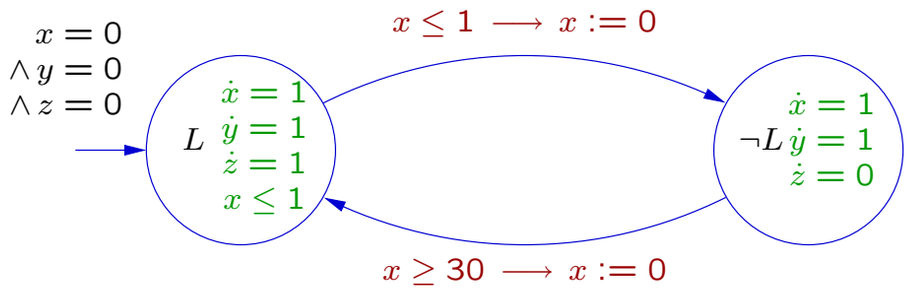
Experiments (1)

The techniques have been implemented as a library of the LASH tool, and used to compute the reachable set of states of several systems including

- Some parametric systems (bakery, Szymanski, Dijkstra, Burns, ...).
- Some communication protocols (alternating-bit protool, ...).
- Some systems manipulating integer variables (lift, ...).
- Some hybrid systems (leaking gas burner, IEEE1394 root contention protocol).

Experiments (2)

Leaking gas burner:



T has 2406 states, $T(A_I)$ has 676 states.

Experiments (3)

Relation	$\lvert T_0\rvert$	$\lvert T_0^*\rvert$	Max $\lvert T_0^i\rvert$
$(x, x+1)$	3	3	11
$(x, x+(1/2))$	9	15	38
$(x, x+(1/7))$	10	40	87
$(x, x+73)$	14	75	933
$(((x,y),(x+2,y-1)) \cup ((x,y),(x-1,y+2)))$ $\cap \mathbf{N}^2 \times \mathbf{N}^2$	19	70	1833
$((x,y),(x+2,y-1)) \cup ((x,y),(x-1,y+2))$ $\cup((x,y),(x+1,y+1))) \cap \mathbf{N}^2 \times \mathbf{N}^2$	21	31	635
$((w,x,y,z),(w+1,x+2,y+3,z+4))$	91	251	2680

Other Applications.

Other applications of (ω-)regular model checking include:

- The verification of heterogenous systems. [Bardin, Finkel, 06].
- The verification of temporal properties [Bouajjani, Legay, Wolper, 04].
- Geometrical applications [Cantin, Legay, Wolper, 07].
- Handling Open Systems [de Alfaro, Faella, Legay 05, 06, 07].

Extensions.

- There are many cases where T^* is not regular.

Solutions:

- Going beyond regular representations [Fisman, Pnueli, 01].

- Using other regular representations

  – Tree Regular Model Checking(see [Abdulla, D'orso, Legay, Rezine, 05, 06] [Bouajjani, Haberhmel, Vojnar, 05, 06]).

  – Applications: communication protocols, XML documents, ...